

Shared vocabulary turns AI ambition into sound project choices

THE WHOLE SYSTEM
A successful AI project designs the full system, not only the model

- 1 Model capability**
Select a model and define what it can see in each interaction.
- 2 Grounded knowledge**
Retrieve trusted, current enterprise information rather than relying on model memory alone.
- 3 Controlled action**
Give the model narrow, observable tools instead of unrestricted system access.
- 4 Measurement and control**
Evaluate quality, enforce guardrails, and monitor cost and latency in production.

Decision requested: Use the vocabulary to frame inputs, monitor evidence, and trade-offs before selecting a solution.

02 A successful AI project designs the full system, not only the model

MODEL FUNDAMENTALS
Tokens and the context window set what the model can process in one interaction

Tokens are model-readable pieces of text
Text is split into tokens before processing; tokenization varies by model and language.

The context window is the model's working set
Instructions, conversation history, retrieved knowledge, and output all compete for the same finite window.

Context is a product design constraint
Decide what to include, summarize, retrieve, or omit so the model sees the relevant evidence.

03 Tokens and the context window set what the model can process in one interaction

MODEL LIFECYCLE
Training builds a model; inference applies it to each request

Training changes model parameters
The learning process adjusts parameters from data to create general capabilities.

Inference applies the trained model
Each request combines an input and context with the existing model to produce an output.

The distinction changes project choices
Most enterprise work improves inference through prompts, retrieval, tools, and controls rather than training a model.

04 Training builds a model; inference applies it to each request

EMBEDDINGS
Embeddings make meaning searchable, and RAG uses that search to ground answers

Prepare knowledge
Chunk approved documents and attach permissions, provenance, and update rules.

Create embeddings
Encode chunks as vectors so related meanings are located near each other.

Retrieve evidence
Match the request to relevant, authorized chunks at run time.

Generate with citations
Give the model retrieved evidence and ask it to answer from that evidence.

05 Embeddings make meaning searchable, and RAG uses that search to ground answers

ADAPTATION CHOICES
Prompting steers a model now, while fine-tuning changes behavior through training

	Prompting Instructions and examples to request data	Fine-tuning Additional training to reprogram insights
Best when	The task or policy changes often	Behavior needs a stable, repeatable pattern
Changes	Input context and instructions	Learned model behavior
Typical project work	Design prompts, templates, retrieval	Curate, label, split, and test a training set
Main governance need	Version prompts and test regressions	Manage data quality, privacy, and evaluation

Verdict: Start with prompting and retrieval; fine-tune only when evidence shows a persistent gap.

06 Prompting steers a model now, while fine-tuning changes behavior through training

REDUCTION LAYERS
Hallucinations are plausible but unsupported outputs, so systems need multiple reduction layers

Ground each answer in approved evidence
Retrieve trusted sources and require citations where users need to verify an answer.

Constrain important outputs
Use structured schemas, validation, and policy checks before a response or action is accepted.

Clarify or escalate ambiguity
Ask for missing context and state high-consequence cases to a qualified person.

Govern the knowledge base
Set provenance, access, freshness, and review rules for the sources used at run time.

07 Hallucinations are plausible but unsupported outputs, so systems need multiple reduction layers

CONTROLLED ACTION
Tool use lets a model act, and function calling supplies the controlled contract

01 Declare allowed tools
Describe each operation with a name, purpose, and typed input schema.

02 Request a structured call
The model selects a tool and produces arguments that match the schema.

03 Validate and execute
The application checks authorization, policy, and inputs before acting.

04 Observe the result
Return a bounded result to the model and log the action for review.

08 Tool use lets a model act, and function calling supplies the controlled contract

AGENTS
Agents repeat plan-act-observe loops, with memory treated as managed state

Plan
Interpret the goal, choose the next step, and define what success looks like.

Act
Generate a response or invoke an approved tool within explicit limits.

Observe
Read the tool result or user feedback and decide whether more work is needed.

Manage memory
Keep only useful state, separate short-term context from durable, governed records.

09 Agents repeat plan-act-observe loops, with memory treated as managed state

QUALITY AND SAFETY
Evaluation and guardrails make quality, safety, and operational control testable

Define success and failure
Build representative loads, expected outcomes, and unacceptable failure modes into an evaluation set.

Test before release
Compare quality, safety, and tool use behavior against agreed acceptance criteria.

Guard at run time
Apply access controls, content policies, schemas, approvals, and human escalation.

Monitor in production
Log outcomes, analyze failures, and refresh evaluations as the product and knowledge change.

10 Evaluation and guardrails make quality, safety, and operational control testable

OPERATIONAL TRADE-OFFS
Capability, cost, and latency form a design trade-off to optimize deliberately

HIGH CAPABILITY, LOW LATENCY
Use the best fit model, allow deeper reasoning and richer context.

HIGH CAPABILITY, HIGH LATENCY
Phrase selectively, cache reusable work, and keep agent loops bounded.

REDUCED, LOW LATENCY
Prefer simpler models and batch work where the experience permits.

REDUCED, HIGH LATENCY
Use narrow tasks, compact context, and fast deterministic controls.

Set a service-level target first, then choose a model, context strategy, and control loop to meet it.

11 Capability, cost, and latency form a design trade-off to optimize deliberately

FROM VOCABULARY TO CHARTER
A short discovery sequence turns the vocabulary into an AI project charter

Next step	Owner	Due
Select a narrow user problem and define the decision or outcome it should improve	Business sponsor	Discovery start
Specify the context, trusted knowledge, tools, and permissions the system may use	Product and engineering	Solution design
Define evaluation cases, failure thresholds, guardrails, and escalation paths	Product and risk owner	Before pilot
Set quality, cost, and latency targets before choosing the production architecture	Technology lead	Before build

12 A short discovery sequence turns the vocabulary into an AI project charter